**UNIVERSIDAD DE GUADALAJARA**

CENTRO UNIVERSITARIO DE CIENCIAS EXACTAS E INGENIERÍA

**SEMINARIO DE SOLUCION DE PROBLEMAS DE SISTEMAS BASADOS EN CONOCIMIENTO**

**Dr. JORGE DE JESUS GALVEZ RODRIGUEZ**

**Sección: D03**

**Reporte de Práctica: Proyecto final**

Logotipo

Descripción generada automáticamente

27/Nov/2024

Ceballos Jiménez Christopher Código: 219750442

Contenido

[Introducción 3](#_Toc183710654)

[Planteamiento del Problema 4](#_Toc183710655)

[Justificación 5](#_Toc183710656)

[Objetivo general 6](#_Toc183710657)

[Metas específicas 6](#_Toc183710658)

[1. Análisis de características: 6](#_Toc183710659)

[2. Construcción del modelo: 6](#_Toc183710660)

[3. Evaluación del desempeño: 6](#_Toc183710661)

[4. Testing con datos reales en Web- app: 6](#_Toc183710662)

[Implementación 7](#_Toc183710663)

[Resultados 18](#_Toc183710664)

[Accuracy: 18](#_Toc183710665)

[Matriz de confusión: 18](#_Toc183710666)

[Reporte de clasificación: 18](#_Toc183710667)

[Conclusiones 19](#_Toc183710668)

[Bibliografía 20](#_Toc183710669)

# Introducción

En el sector financiero, la evaluación de riesgos es una actividad clave para la toma de decisiones relacionadas con la aprobación de créditos. **Identificar si un solicitante representa un riesgo alto o bajo permite a las instituciones minimizar pérdidas, optimizar recursos y ofrecer soluciones financieras más responsables. Sin embargo, este proceso puede ser complejo debido a la gran cantidad de variables involucradas, como el historial crediticio, los ingresos, las deudas actuales y los patrones de pago del cliente.**

El presente reporte aborda este problema mediante el uso de técnicas de machine learning, específicamente el algoritmo ***Random Forest Classifier***. Este modelo, conocido por su capacidad para manejar datos no lineales y su robustez frente a valores atípicos, será empleado para predecir la probabilidad de riesgo asociado con solicitudes de crédito. El análisis se basará en un conjunto de datos que contiene información histórica y demográfica de clientes, la cual será utilizada para entrenar, validar y evaluar el desempeño del modelo.

Este enfoque pretende demostrar cómo el uso de herramientas avanzadas de análisis de datos puede mejorar los procesos tradicionales de evaluación crediticia, contribuyendo a la eficiencia operativa y la reducción de riesgos en el sector financiero.

# Planteamiento del Problema

En el ámbito financiero, la evaluación de riesgo crediticio es un proceso crítico que impacta directamente en la rentabilidad y sostenibilidad del negocio. Decidir si otorgar un crédito conlleva analizar múltiples variables de los solicitantes, como historial de pagos, ingresos y nivel de endeudamiento. **Tradicionalmente, este análisis se realiza mediante reglas predefinidas y modelos estadísticos básicos, lo que puede ser insuficiente para capturar patrones complejos y no lineales en los datos.**

**Desde una perspectiva de negocio, un modelo ineficiente o inexacto puede resultar en dos tipos de errores: aprobar créditos de alto riesgo que incrementan las tasas de morosidad y pérdidas financieras, o rechazar solicitudes de bajo riesgo que representan oportunidades de ingresos.** En un mercado competitivo, este tipo de decisiones inadecuadas puede afectar tanto la rentabilidad como la percepción de los clientes sobre la institución.

Desde el enfoque de ingeniería de software, el desafío radica en diseñar e implementar una solución tecnológica escalable, precisa y eficiente que automatice la clasificación del riesgo crediticio. Esta solución debe integrarse de manera fluida con los sistemas existentes, manejar grandes volúmenes de datos y permitir un procesamiento en tiempo real para responder a las necesidades operativas del negocio.

Por ello, este proyecto plantea el desarrollo de un modelo predictivo basado en técnicas de machine learning, específicamente ***Random Forest Classifier***. Este modelo no solo **permite mejorar la precisión en la predicción del riesgo crediticio, sino que también facilita la interpretabilidad de los resultados y la toma de decisiones informadas**. Al combinar ingeniería de software con análisis avanzado de datos, se busca crear una herramienta robusta que transforme el proceso de evaluación de crédito, optimizando tanto los tiempos de procesamiento como la calidad de las decisiones.

Esta iniciativa tiene como objetivo resolver un problema técnico y de negocio clave***: maximizar la eficiencia operativa al tiempo que se minimizan los riesgos financieros, contribuyendo al crecimiento sostenible de la financiera .***

# Justificación

La evaluación precisa del riesgo crediticio es fundamental para la ***sostenibilidad y rentabilidad de las instituciones financieras***. Los métodos tradicionales suelen ser insuficientes para manejar la complejidad de los datos actuales, lo que incrementa el riesgo de pérdidas financieras por morosidad o el rechazo de oportunidades de negocio.

El uso de técnicas de machine learning, como el ***Random Forest Classifier***, permite desarrollar modelos predictivos más precisos, dinámicos y escalables***, optimizando la toma de decisiones y reduciendo costos operativos.*** Este enfoque no solo mejora la eficiencia operativa, sino que también fortalece la competitividad de la institución al ofrecer procesos más ágiles y confiables, alineados con las demandas del mercado moderno.

# Objetivo general

El presente proyecto tiene como propósito principal ***abordar el problema de la evaluación de riesgo crediticio mediante la aplicación de técnicas avanzadas de machine learning***. Específicamente, se busca construir un modelo predictivo basado en el algoritmo ***Random Forest Classifier*** que permita clasificar a los solicitantes de crédito como de alto o bajo riesgo. Este modelo se desarrollará a partir de un análisis detallado de características financieras y demográficas, empleando un conjunto de datos obtenido de Kaggle.

## Metas específicas

1. Análisis de características: Examinar y procesar las variables del dataset con el fin de identificar cuáles tienen mayor influencia en la predicción del riesgo crediticio. Esto incluye la limpieza, transformación, EDA y selección de datos relevantes para el modelo.

2. Construcción del modelo: Implementar un modelo de machine learning utilizando el algoritmo ***Random Forest Classifier***, aprovechando su capacidad para manejar datos no lineales, reducir el sobreajuste y ofrecer interpretabilidad en los resultados.

3. Evaluación del desempeño: Validar el modelo mediante métricas estándar como ***precisión, recall, y F1-score,*** asegurando que los resultados sean consistentes y confiables para la clasificación del riesgo.

4. Testing con datos reales en Web- app: Diseñar una web-app que facilite la implementación del modelo en un entorno práctico, permitiendo su actualización y uso recurrente conforme se disponga de nuevos datos.

# Implementación

# %%

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import LabelEncoder, StandardScaler

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score

# %% [markdown]

# # 2. Cargar csv y analizar datos generales 🆙

#

# Verificamos la info del dataset y visualizamos las primeras lineas

# %%

# Cargamos el dataset

df = pd.read\_csv('./credit\_data.csv')

#dropear el index

df = df.iloc[:, 1:] # Conserva todas las columnas excepto la primera (posición 0)

# ver la info general

df.info()

# Ejemplo con las primeras lineas

df.head()

# %% [markdown]

# # 3. Pre-procesamiento de los datos 🏭

#

# - Pasar de Marcos alemanes a MXN

# - Necesitamos manejar los valores faltantes y codificar las variables categóricas en formatos numéricos.

# %%

# Manejar los NA con un placeholder

df = df.copy()

#Conversion de Marcos alemanes a MXN: tipo de cambio 10.88 al 25/Nov/2024

def dm\_a\_mxn(df, nombre\_columna):

# Convertir la columna a numérica, forzando errores a NaN

df[nombre\_columna] = pd.to\_numeric(df[nombre\_columna], errors='coerce')

# Multiplicar solo los valores numéricos

df[nombre\_columna] = df[nombre\_columna] \* 10.88

return df

df = dm\_a\_mxn(df,'Credit amount') #Monto del crédito

# %%

df['Saving accounts'] = df['Saving accounts'].fillna('unknown')

df['Checking account'] = df['Checking account'].fillna('unknown')

"""

Utiliza un bucle para procesar varias columnas categóricas: Sex, Housing, Saving accounts, Checking account, y Purpose.

Usa LabelEncoder para transformar estas categorías en valores numéricos:

Ejemplo: "male" -> 1, "female" -> 0.

Guarda los objetos LabelEncoder en un diccionario (label\_encoders).

"""

label\_encoders = {}

encoded\_mappings = {}

for column in ['Sex', 'Housing', 'Saving accounts', 'Checking account', 'Purpose']:

le = LabelEncoder()

df[column] = le.fit\_transform(df[column])

label\_encoders[column] = le

# Guardar el mapeo original -> codificado

encoded\_mappings[column] = dict(zip(le.classes\_, range(len(le.classes\_))))

df.head()

# %%

for column, mapping in encoded\_mappings.items():

print(f"Column: {column}")

for original, encoded in mapping.items():

print(f" {original} -> {encoded}")

# output\_file = "label\_encodings.txt"

# with open(output\_file, "w") as file:

# for column, mapping in encoded\_mappings.items():

# file.write(f"Column: {column}\n")

# for original, encoded in mapping.items():

# file.write(f" {original} -> {encoded}\n")

# file.write("\n") # Espacio entre columnas

# %% [markdown]

# # 4. Exploratory Data Analysis (EDA) 🤯

#

# Analisis de las features para saber su relación con el target y su distribución

# %%

# Convertir los datos infinitos a NaNs

df = df.replace([np.inf, -np.inf], np.nan)

# Crear el histograma con Seaborn

hist = sns.histplot(df['Age'], kde=True, bins=20, edgecolor='black')

# Agregar gradiente a las barras

norm = plt.Normalize(hist.patches[0].get\_height(), max(p.get\_height() for p in hist.patches))

for patch in hist.patches:

color = plt.cm.viridis(norm(patch.get\_height())) # Usar el colormap 'viridis'

patch.set\_facecolor(color)

# Configurar el título y mostrar el gráfico

plt.title('Distribución de Edad')

plt.xlabel('Edad')

plt.ylabel('Frecuencia')

plt.show()

# %%

#Crear el histograma con Seaborn

hist2 = sns.histplot(df['Credit amount'], kde=True, bins=20, edgecolor='black')

# Agregar gradiente a las barras

norm = plt.Normalize(hist2.patches[0].get\_height(), max(p.get\_height() for p in hist2.patches))

for patch in hist2.patches:

color = plt.cm.plasma(norm(patch.get\_height())) # Usar el colormap 'plasma'

patch.set\_facecolor(color)

# Agregar texto (valor de frecuencia) sobre las barras

height = patch.get\_height()

width = patch.get\_width()

x = patch.get\_x() + width / 2 # Centrado en la barra

y = height

# Colocar el valor sobre la barra con un desplazamiento en y para que no se sobreponga

plt.text(x, y, f'{int(height)}', ha='center', va='bottom', fontsize=10)

# Configurar el título y mostrar el gráfico

plt.title('Distribución del Monto del Crédito solicitado')

plt.xlabel('Monto del Crédito')

plt.ylabel('Frecuencia')

plt.show()

# %%

# Heatmap de correlacion

plt.figure(figsize=(10, 8))

sns.heatmap(df.corr(), annot=True, cmap='coolwarm')

plt.title('Heatmap de correlacion de Features')

plt.show()

# %% [markdown]

# # 5. Separar entre train set y test set 🚂 / 🧾

# Dividimos los datos en características (X) y objetivo (y), luego en conjuntos de entrenamiento y pruebas.

# %%

# Definir features y target

X = df.drop(columns=['Credit amount']) # Tomamos Credit amount como variable target

# Clasificacion binaria (credito solicitado Bajo / Alto), si > a la media

y = df['Credit amount'] > df['Credit amount'].median()

# %%

# Separar entre en trainning set y el testing set

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# # Standardize the features

# scaler = StandardScaler()

# X\_train = scaler.fit\_transform(X\_train)

# X\_test = scaler.transform(X\_test)

# %% [markdown]

# ¿Es necesaria la estandarización para Random Forest?

# \*\*No, estrictamente no es necesario\*\*

#

# Sin embargo:

#

# Si las características tienen rangos extremadamente dispares (por ejemplo, edad de 20-70 y montos de crédito de 25,000-200,000), estandarizar ayuda a que el modelo aprenda de manera más uniforme durante la selección de atributos.

#

# Este proceso transforma los datos para que tengan una distribución con una media (promedio) de 0 y una desviación estándar de 1. Esto se hace restando la media de cada dato y luego dividiendo por la desviación estándar del conjunto de datos original.

#

# ![image.png](attachment:image.png)

# %% [markdown]

# # 6. Fase de entrenamiento

# Se usará un modelo clasificador tipo Random forest

# %% [markdown]

# # 🦾 Etapa de entrenamiento 🦾

#

# %%

model = RandomForestClassifier(n\_estimators=10000, random\_state=69420, verbose=1)

model.fit(X\_train, y\_train)

# %% [markdown]

# # 🏆 Etapa de Prediccion 🏆

# %%

X\_train

# %%

# Predicciones en el set de testeo

y\_pred = model.predict(X\_test)

# %% [markdown]

# # 7. 📝Evaluamos el modelo:

# Evaluamos el rendimiento del modelo mendiate:

# 1. La precisión,

# 2. La matriz de confusión

# 3. El informe de clasificación.

#

# %%

accuracy = accuracy\_score(y\_test, y\_pred)

print(f"Accuracy: {accuracy:.2f}")

print("------------------------------")

# %% [markdown]

# ### Explicación:

#

# - \*\*Exactitud (Accuracy):\*\*

# El modelo alcanzó una exactitud del 76%, lo que significa que predijo correctamente el estado de riesgo crediticio en el 76% de las muestras de prueba.

# %%

conf\_matrix = confusion\_matrix(y\_test, y\_pred)

print("Matriz de confusión:")

print(conf\_matrix)

# %% [markdown]

#

# - \*\*Matriz de confusión:\*\*

# La matriz de confusión muestra cuántas instancias reales de cada clase fueron clasificadas correcta o incorrectamente.

# - \*\*Verdaderos Negativos (120):\*\* El modelo identificó correctamente a 120 personas de bajo riesgo.

# - \*\*Falsos Positivos (35):\*\* El modelo clasificó incorrectamente a 35 personas de alto riesgo como de bajo riesgo.

# - \*\*Falsos Negativos (38):\*\* El modelo clasificó incorrectamente a 38 personas de bajo riesgo como de alto riesgo.

# - \*\*Verdaderos Positivos (107):\*\* El modelo identificó correctamente a 107 personas de alto riesgo.

#

# %%

class\_report = classification\_report(y\_test, y\_pred)

print("------------------------------")

print("Reporte de clasificacion:")

print(class\_report)

# %% [markdown]

# - \*\*Informe de clasificación:\*\*

# - \*\*Precisión (Precision):\*\* Mide la exactitud de las predicciones positivas.

# - \*\*Sensibilidad (Recall):\*\* Mide la capacidad de identificar instancias positivas.

# - \*\*Puntaje F1 (F1-Score):\*\* Media armónica de precisión y sensibilidad, útil para evaluar conjuntos de datos desbalanceados.

#

# ### El modelo muestra un rendimiento equilibrado entre las dos clases, aunque aún hay margen de mejora.

# %% [markdown]

# # Conclusión:

#

#

# - El modelo alcanzó una exactitud del 76%, lo que indica un desempeño razonable en la predicción del riesgo crediticio.

# - La matriz de confusión y el informe de clasificación sugieren que, aunque el modelo tiene un buen desempeño general, todavía existen errores de clasificación, especialmente en los falsos positivos y falsos negativos.

# %% [markdown]

# # 2NDA PARTE-----------------------------

# ## Testing en produccion

# %% [markdown]

# >Pasamos a `.pkl` para serializar el modelo y usarlo como un archivo

# %%

import pickle

# Guardar el modelo con pickle

with open("random\_forest\_model.pkl", "wb") as file:

pickle.dump(model, file)

print("Modelo guardado en 'random\_forest\_model.pkl'")

# %%

import joblib

# Guardar el modelo entrenado

joblib.dump(model, 'joblib\_random\_forest\_model.pkl')

# ## 🐲Si quieres contribuir o descargar el codigo:

#

# ## https://github.com/Chelqq/Credit-risk-classifier

# Resultados

### Accuracy:

Texto

Descripción generada automáticamente

El modelo alcanzó una precisión del 76%, lo que significa que predijo correctamente el estado de riesgo crediticio en el 76% de las muestras de prueba.

### Matriz de confusión:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente con confianza media

* Verdaderos Negativos (120): El modelo clasificó correctamente a 120 personas como de bajo riesgo, lo que indica que estas predicciones coincidieron con los valores reales de la clase "bajo riesgo".
* Falsos Positivos (35): En 35 casos, el modelo clasificó incorrectamente a personas de alto riesgo como si fueran de bajo riesgo, lo que representa un error crítico desde el punto de vista financiero, ya que podría resultar en la aprobación de créditos de alto riesgo.
* Falsos Negativos (38): Hubo 38 casos en los que el modelo clasificó a personas de bajo riesgo como si fueran de alto riesgo. Aunque este error minimiza el riesgo financiero, puede llevar a la pérdida de oportunidades de negocio.
* Verdaderos Positivos (107): El modelo identificó correctamente a 107 personas como de alto riesgo, mostrando su efectividad para detectar clientes con una alta probabilidad de incumplimiento.

### Reporte de clasificación:

Calendario

Descripción generada automáticamenteEstas métricas indican que el modelo logra un desempeño equilibrado entre ambas clases, con una ligera superioridad en la identificación de casos de bajo riesgo. Sin embargo, los errores representados por los falsos positivos y negativos tienen implicaciones financieras importantes, ya que podrían traducirse en otorgamiento de créditos riesgosos o en rechazos a solicitantes confiables.

# Conclusiones

El desarrollo de este proyecto de clasificación para la evaluación del riesgo crediticio utilizando un modelo de **Random Forest Classifier permitió abordar un problema fundamental en el sector financiero: identificar a los solicitantes de crédito con alto o bajo riesgo de incumplimiento.** A través del análisis y modelado de datos financieros y demográficos, se lograron construir predicciones que pueden facilitar la toma de decisiones estratégicas en la concesión de créditos.

El modelo alcanzó una precisión global del **76%**, demostrando un desempeño aceptable en términos de clasificación. Las métricas de evaluación revelaron un equilibrio en la identificación de casos de alto y bajo riesgo, aunque con oportunidades claras de mejora. La matriz de confusión mostró que el modelo logró identificar correctamente la mayoría de los casos de bajo riesgo (120 verdaderos negativos) y alto riesgo (107 verdaderos positivos). Sin embargo, los falsos positivos y negativos (35 y 38 casos, respectivamente) subrayan la necesidad de seguir afinando el modelo, ya que estos errores podrían tener consecuencias financieras importantes, como la aprobación de créditos riesgosos o el rechazo de solicitantes confiables.

Una de las limitaciones identificadas fue la presencia de valores faltantes en algunas características del dataset (Saving accounts y Checking account), lo que pudo haber afectado la capacidad predictiva del modelo. Esto destaca la importancia de una preparación exhaustiva de los datos antes del entrenamiento del modelo, incluyendo estrategias de rellenado o eliminación de valores ausentes.

Este proyecto demuestra la aplicabilidad de las herramientas de Machine Learning para resolver problemas complejos en el sector financiero, proporcionando insights valiosos que pueden traducirse en decisiones más informadas y efectivas. No obstante, se requiere un esfuerzo continuo para refinar el modelo y adaptarlo a escenarios reales, con el objetivo de maximizar su impacto y confiabilidad en el contexto operativo de una institución financiera.

# Bibliografía

German Credit Risk. (2016, 14 diciembre). Kaggle. https://www.kaggle.com/datasets/uciml/german- credit

Kabure. (2018, 3 noviembre). Predicting Credit Risk model pipeline. Kaggle.

https://www.kaggle.com/code/kabure/predicting-credit-risk-model-pipeline

Carbon. (s. f.). Carbon. https://carbon.now.sh/

German credit risk with target. (2018, 9 enero). Kaggle.

https://www.kaggle.com/datasets/kabure/german-credit-data-with-risk

Normalized Nerd. (2021, 21 abril). Random Forest algorithm clearly explained! [Vídeo]. YouTube. https://www.youtube.com/watch?v=v6VJ2R066Ag

Normalized Nerd. (2021a, enero 13). Decision tree classification clearly explained! [Vídeo]. YouTube. https://www.youtube.com/watch?v=ZVR2Way4nwQ

Ibm. (2024, 8 octubre). Arboles de decisión. IBM. https://www.ibm.com/es-es/topics/decision-trees

Ibm. (2024b, octubre 25). Random Forest. IBM. https://www.ibm.com/topics/random-forest